AI-based Heart Disease Prediction and Time Series Forecasting

System Documentation

Somesh Gupta (20817245)
Ankita Kapoor (20811694)
Michael Lynch (20852964)
Shadman Raihan (20858688)
Nitheesha Reddy Penta Reddy (20811504)

Overview

- This project is a web application that uses machine learning to analyze the UCI heart disease dataset and make predictions based on supplied medical data.
- The website should display the risk factors to the user as well as a probability or confidence score of their health in the coming years. If possible, the lifestyle choices to be made should also be displayed.

Link to our Website: http://35.196.37.255/

DEVELOPMENT

For development, we followed the Agile Model and attempted to incorporate continuous integration and deployment. We held weekly meetings and kept in touch using the Slack IM platform, as well as Skype calls when necessary. Tasks were initially divided up using Basecamp although towards the end of the project we started using Gitlab issues and milestones to track progress.

Testing was initially done manually, although automated tests were eventually written in Pytest, with Docker and Docker-Compose being used to set up the test environment. A completely automated CI/CD pipeline was attempted but was unsuccessful due to architectural choices made early in the project. The benefits of test-driven development became apparent towards the end of the project and in retrospect would have been much easier to implement at the beginning.

For version control, we used git and synced our project with a repository hosted on the University's GitLab server. The protected master branch was used for the most recent stable version to be deployed, a 'dev' branch was set to the most recent development awaiting testing and various subbranches were created for individual tasks. This approach ensured that the code was developed in an organized fashion and always had a working instance available to quickly revert to.

There are 3 main modules in our project which are as follows:

1) Machine Learning:

❖ Data:

The dataset is an aggregate of the following raw files-

1. Cleveland Clinic Foundation (cleveland.data)
2. Hungarian Institute of Cardiology, Budapest (hungarian.data)
3. V.A. Medical Center, Long Beach, CA (long-beach-va.data)
4. University Hospital, Zurich, Switzerland (switzerland.data)

Total: 1000 Data-points
Features: 76 (including the target variable)
Target Variable: Goal [it is integer-valued from 0 (no presence) to 4]

The data was initially accessed on Kaggle although a more complete instance of the dataset was found on the UCI server at https://archive.ics.uci.edu/ml/datasets/Heart+Disease. The data was initially in a raw format with no labels, although text files with documentation were available which enabled cleaning and reconstruction. Formatting errors in the dataset resulted in about 70 data points being unusable, and these were discarded.
The data was loaded into a pandas data frame format and manually labeled, then the four datasets were merged into one and the data was saved in CSV format for portability.
The labeling script also printed out information on the dataset, such as further explanations of the labels.

After feature selection we got:

Features: **15** ['cp', 'thalach', 'oldpeak', 'rldv5e', 'ladprox', 'laddist', 'cxmain', 'ramus', 'om1', 'rcaprox', 'rcadist','age','sex','chol','target'] (including the target variable)

Attribute documentation:

age
sex
chol: `serum cholestoral in mg/dl`
cp: `-- Value 0: no chest pain`

```
 -- Value 1: typical angina
 -- Value 2: atypical angina
 -- Value 3: non-anginal pain
```

-- Value 4: asymptomatic

thalach: maximum heart rate achieved

oldpeak: ST depression induced by exercise relative to rest

rldv5e: height at peak exercise

ladprox: Proximal Left Anterior Descending Artery

laddist: Distal Left Anterior Descending Artery

cxmin: Main Circumflex Artery

ramus: Ramus Intermedius Coronary Artery

om1: First Obtuse Marginal

rcaprox: Proximal Right Coronary Artery

rcadist: Distal Right Coronary Artery

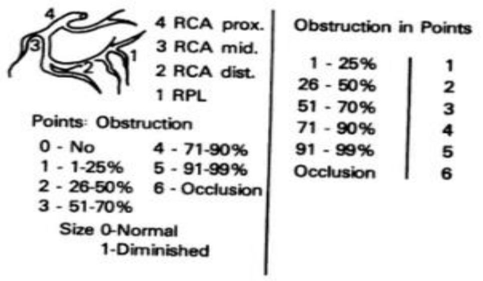


FIGURE 1

Evaluation of coronary angiographic data.

RCA prox. = Proximal right coronary artery
RCA mid. = Middle right coronary artery
RCA dist. = Distal right coronary artery
RPL = Right posterior lateral artery.

Target Variable: Target [Class 0= Low Risk of Heart disease and Class 1= High risk of Heart disease] after mapping 2,3,4 values to 1 in the target from those 4 datasets.

❖ Tools and Technologies:

Libraries such as - NumPy, pandas, Scikit-Learn, Matplotlib, seaborn

❖ Process followed:

1) In our dataset, we have the target variable that is fairly distributed with the count of around 400 for 'people with no heart disease' and 600 for 'people with heart disease'

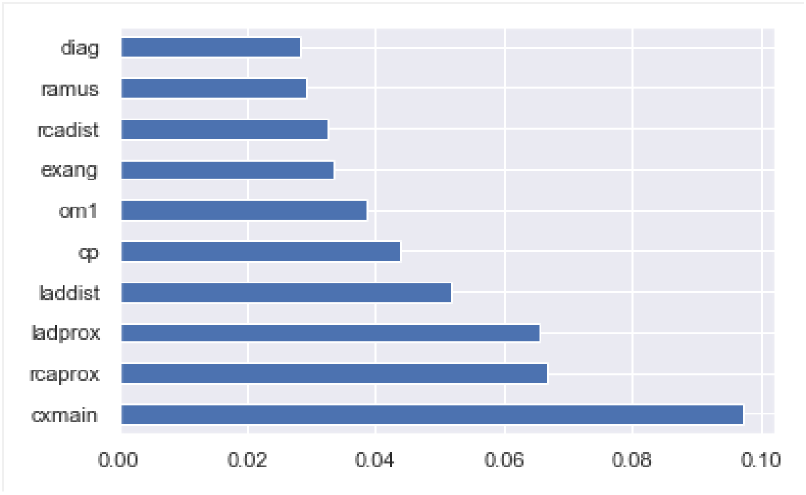
2) Feature Selection-

- Split data into train 80% and test 20%.

- Applied ‘Random Forest Classifier’ with n_estimators = 10

3) Feature Importance-

- We can get the feature importance of each feature of your dataset by using the feature importance property of the model.
- Feature importance gives us a score for each feature of your data, the higher the score more important or relevant is the feature towards our output variable.
- Feature importance is an inbuilt class that comes with Tree-Based Classifiers, we used Extra Tree Classifier for extracting the top 10 features for the dataset.



Plot graph of feature importances for better visualization

4) Recursive feature elimination (RFE) with random forest-

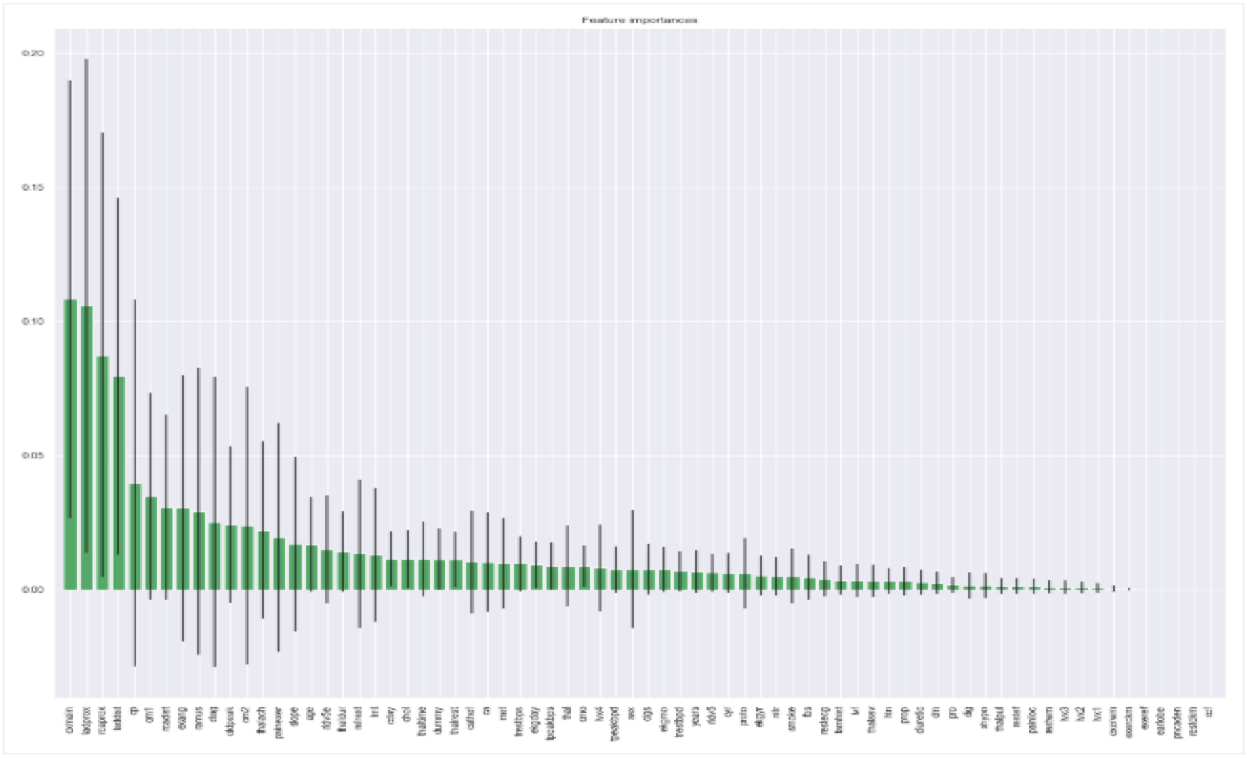
- It uses one of the classification methods (random forest in our example), assigns weights to each of the features whose absolute weights are the smallest are pruned from the current set features. That procedure is recursively repeated on the pruned set until the desired number of features.
- Choosing the top 10 features using the RFE method.

5) Recursive feature elimination with cross-validation and random forest-

- Finding the optimal number of features using the RFE with cross-validation.
- Output:
  Optimal number of features : 11
  Best features : Index(['cp', 'thalach', 'oldpeak', 'rldv5e', 'ladprox', 'laddist', 'cxmain', 'ramus', 'om1', 'rcaprox', 'rcadist'], dtype='object')

6) Tree-based feature selection and random forest classification-

- In the random forest classification method, there are a feature important attributes that are the feature importances (the higher, the more important the feature). To use the feature_importance method, in training data there should not be correlated features. The random forest chooses randomly at each iteration, therefore the sequence of feature importance list can change.

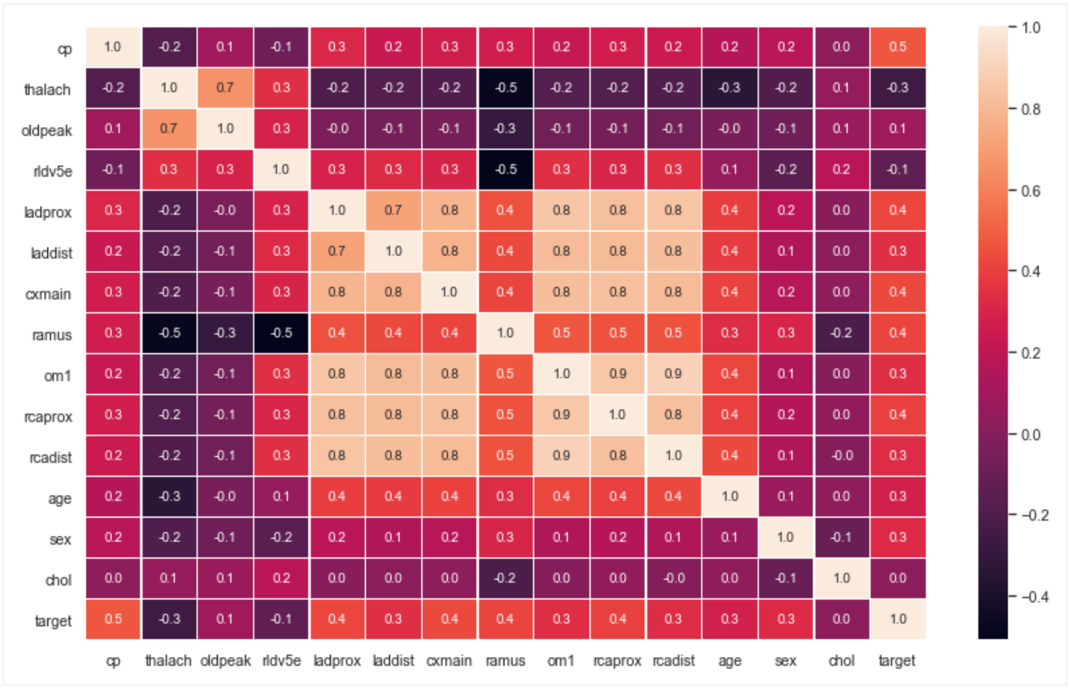


Feature Ranking Curve

7) Feature Extraction-

- We used PCA, Kernal PCA as well as t-SNE but we were not able to segregate our data points.
- So instead of this we selected few important features from all the other above mentioned analysis. Those 15 features including the target are: [['cp', 'thalach', 'oldpeak', 'rldv5e', 'ladprox', 'laddist', 'cxmain', 'ramus', 'om1', 'rcaprox', 'rcadist','age','sex', 'chol','target']

Out [55]:



Correlation(heatMap) for the selected features

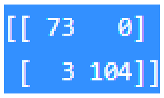
8) Classification Algorithms:

- On these final set of features, we applied 4 classification algorithms and got the following accuracy scores:

Algo	Acc. Score(in %)
Random Forest	98.33
KNN	67.22
Decision Tree	96.67
Naïve Bayes	70

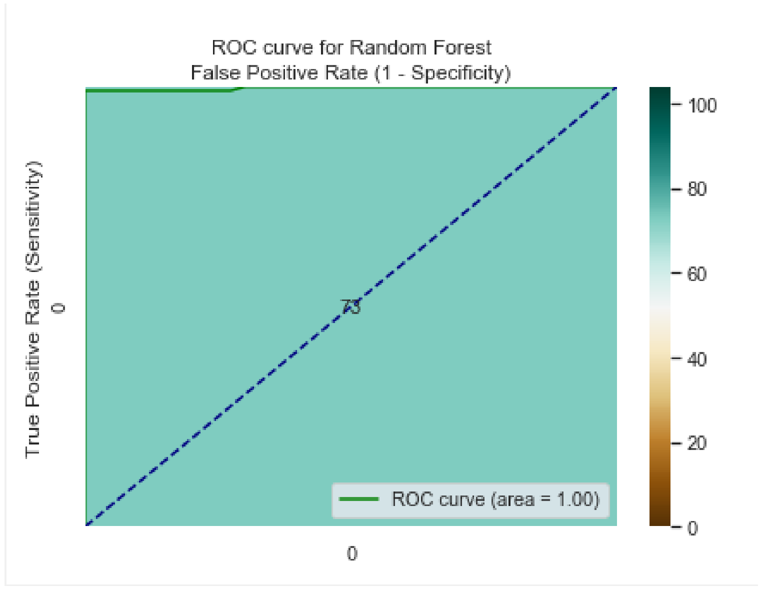
- We also applied a boosting technique i.e AdaBoostClassifier which gave the same accuracy as obtained by the Random Forest which is assumed to be the best.
- So we chose the Random Forest classifier for our model which gave the highest accuracy of 98.33%.

Confusion Matrix for Random Forest:



```
[[ 73   0]
 [  3 104]]
```

ROC curve for Random forest:



9) Time Series: [Future Work]

Initially, our idea was when a patient enters his data into the website, our website should predict the risk of having heart disease as well as forecast his/her future risk based on the data he/she has entered. When we have finished the analysis, to forecast a future, the history of the patient is needed but the UCI website doesn't have the history of the data. So we have pushed this concept for future work and is still in the development phase. Currently, when the same patient enters his data, his time series of data is stored in the database. From that data provided by the user, we have forecasted their future using ARIMA/SARIMA model for now which is a continuous process.

2) <u>Back-end and Integration:</u>

❖ Tools and Technologies:

For Backend: Flask tool is used which is a micro-web framework→libraries used are render_template, Requests, URL_for, Redirect
For Database: Pymongo is used which is a server-based database.
For Integration: used Docker
For Deployment: used the Amazon and Google cloud platforms (AWS and GCP)

❖ Process Followed:

Back-end: We created a Landing page in which we implemented the fields. Then a database is created. Data given by the user is stored in the database and that data from the database is fetched and provided to the machine learning model. The model then returns a value on which the prediction is done. The value returned can be either 0 or 1, where 0 denotes that 'the user is having a low risk of heart disease' and 1 denotes that 'the user is having a high risk of heart disease''

Integration:

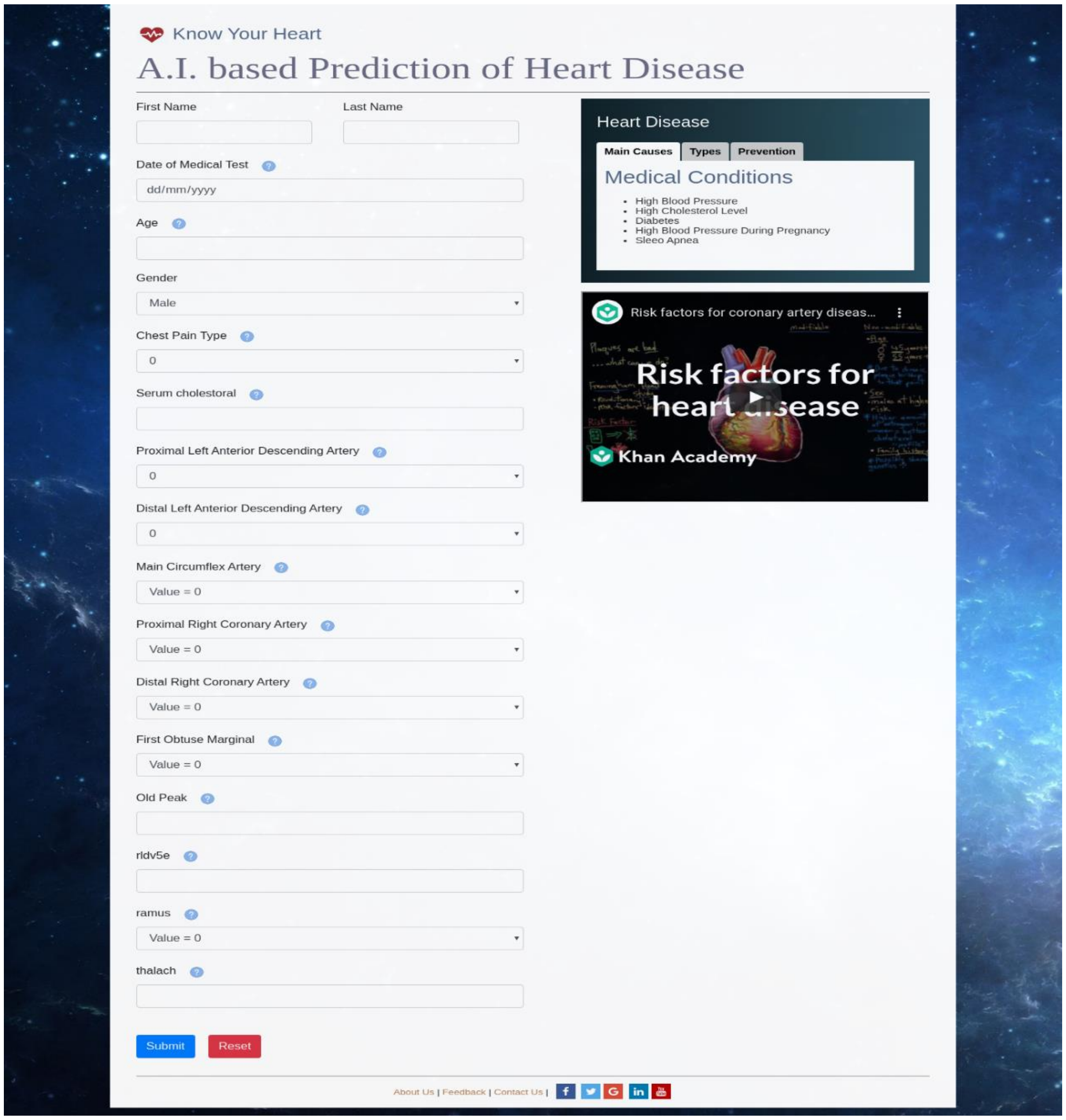
- We used Docker to containerize the components of the web application and docker-compose for orchestration Containerizing the application allowed a standard development environment and easy integration with cloud platforms for deployment.
- An attempt to deploy the web application using Kubernetes on GKE was unsuccessful due to configuration errors between GitLab and Google. The error was not resolved before the end of the project and would likely need to be debugged by starting from scratch.
- The deployment on GKE integrated with a GitLab build pipeline which reported on the status of the project after each push to the remote repository although due to the previously mentioned errors, this did not work. However, transitioning to a full CI/CD model would have made the project proceed much more quickly.

3) <u>Front-end:</u>

❖ **Tools and technologies:** HTML, CSS, JAVASCRIPT, BOOTSTRAP

❖ **Process Followed:**
User Interface: There are three main pages:

1) **Home page/ Landing page:**
Fields: On this page, the user will enter his details. There are 17 fields in total which all are mandatory. Out of 17 fields, 14 are the important features found from the data analysis and the other 3 are just the basic information which includes First Name, Last Name, and Date on which the user enters the details and checks his result.



Know Your Heart

A.I. based Prediction of Heart Disease

First Name
Last Name
Date of Medical Test: dd/mm/yyyy
Age
Gender: Male
Chest Pain Type: 0
Serum cholestoral
Proximal Left Anterior Descending Artery: 0
Distal Left Anterior Descending Artery: 0
Main Circumflex Artery: Value = 0
Proximal Right Coronary Artery: Value = 0
Distal Right Coronary Artery: Value = 0
First Obtuse Marginal: Value = 0
Old Peak
rldv5e
ramus: Value = 0
thalach

Heart Disease

Main Causes | Types | Prevention

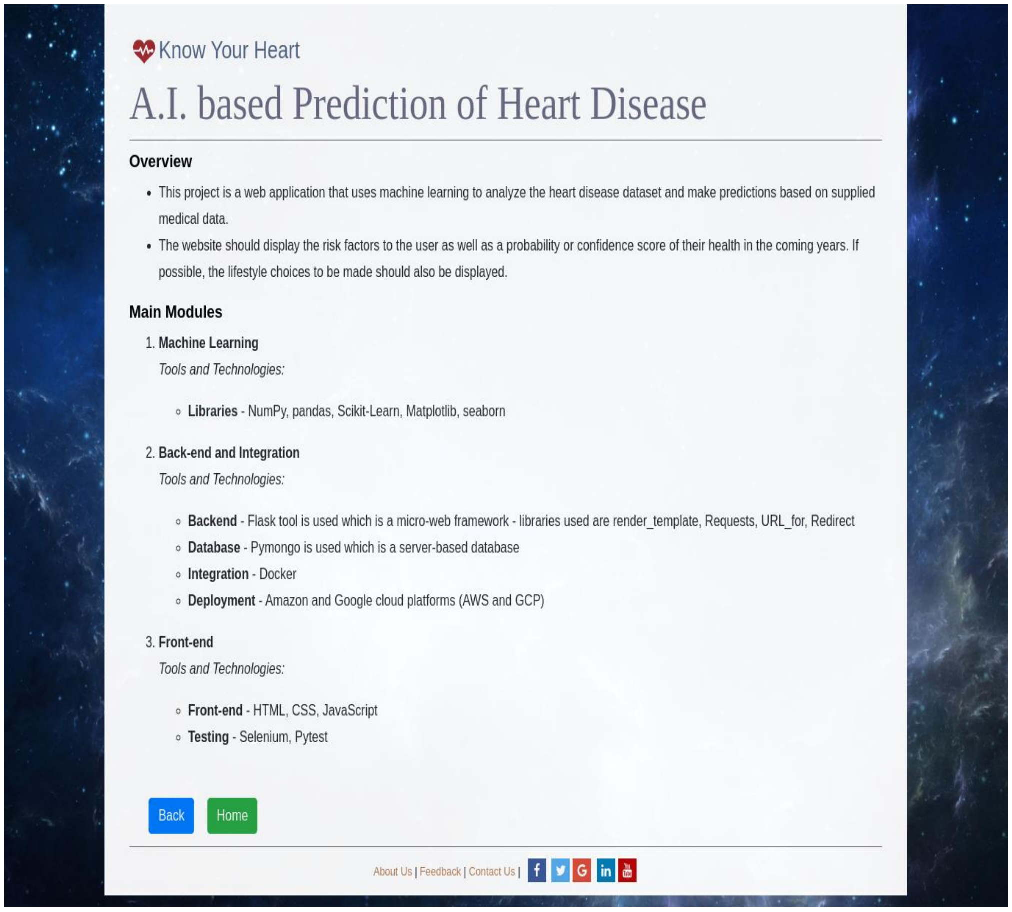
Medical Conditions

- High Blood Pressure
- High Cholesterol Level
- Diabetes
- High Blood Pressure During Pregnancy
- Sleeo Apnea

Submit
Reset
Risk factors for coronary artery diseas...
Risk factors for heart disease
Khan Academy
About Us | Feedback | Contact Us |

There are three links (About Us | Feedback | Contact Us) on the footer as well as social media links four our website.

About Us page: Describes the overview of our website.



Know Your Heart

A.I. based Prediction of Heart Disease

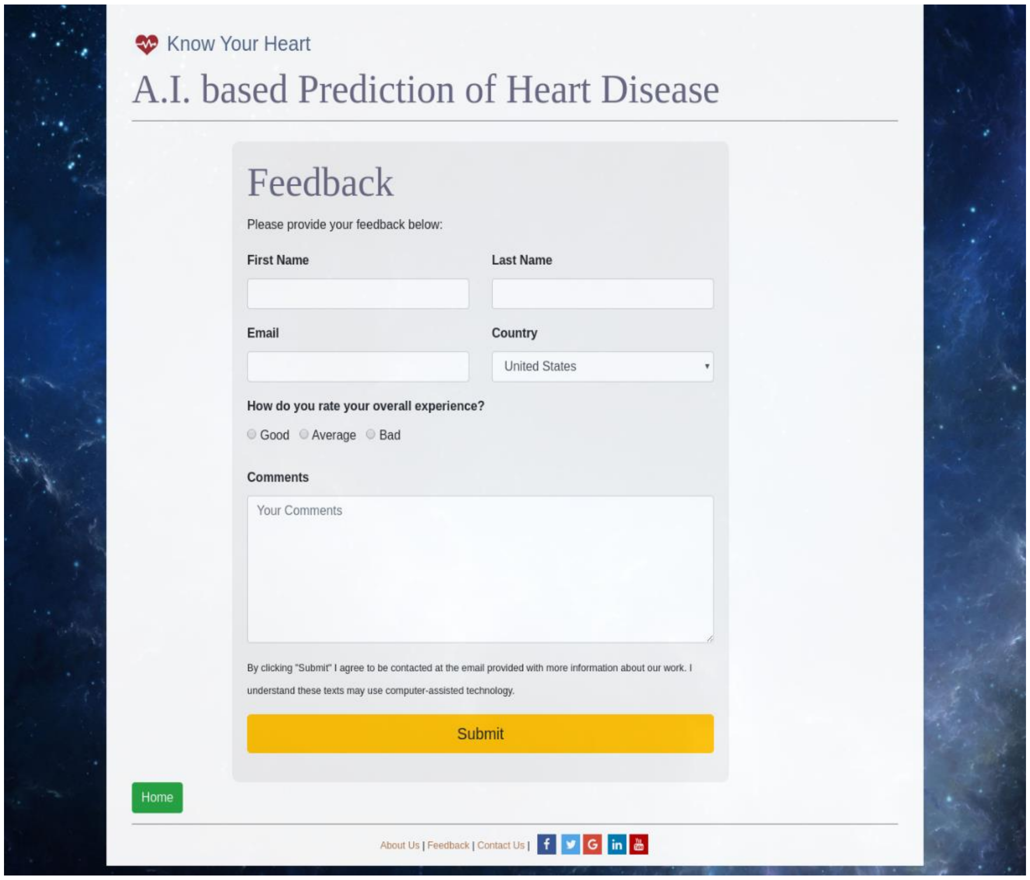
Overview

- This project is a web application that uses machine learning to analyze the heart disease dataset and make predictions based on supplied medical data.
- The website should display the risk factors to the user as well as a probability or confidence score of their health in the coming years. If possible, the lifestyle choices to be made should also be displayed.

Main Modules

1. **Machine Learning**
   Tools and Technologies:
   - **Libraries** - NumPy, pandas, Scikit-Learn, Matplotlib, seaborn
2. **Back-end and Integration**
   Tools and Technologies:
   - **Backend** - Flask tool is used which is a micro-web framework - libraries used are render_template, Requests, URL_for, Redirect
   - **Database** - Pymongo is used which is a server-based database
   - **Integration** - Docker
   - **Deployment** - Amazon and Google cloud platforms (AWS and GCP)
3. **Front-end**
   Tools and Technologies:
   - **Front-end** - HTML, CSS, JavaScript
   - **Testing** - Selenium, Pytest

Feedback page: Where users can give feedback related to our work/website. There are certain fields that he has to enter while giving feedback which includes the Overall Rating and Comments section as well.


Know Your Heart

A.I. based Prediction of Heart Disease

Feedback

Please provide your feedback below:

First Name

Last Name

Email

Country

United States

How do you rate your overall experience?

Good Average Bad

Comments

Your Comments

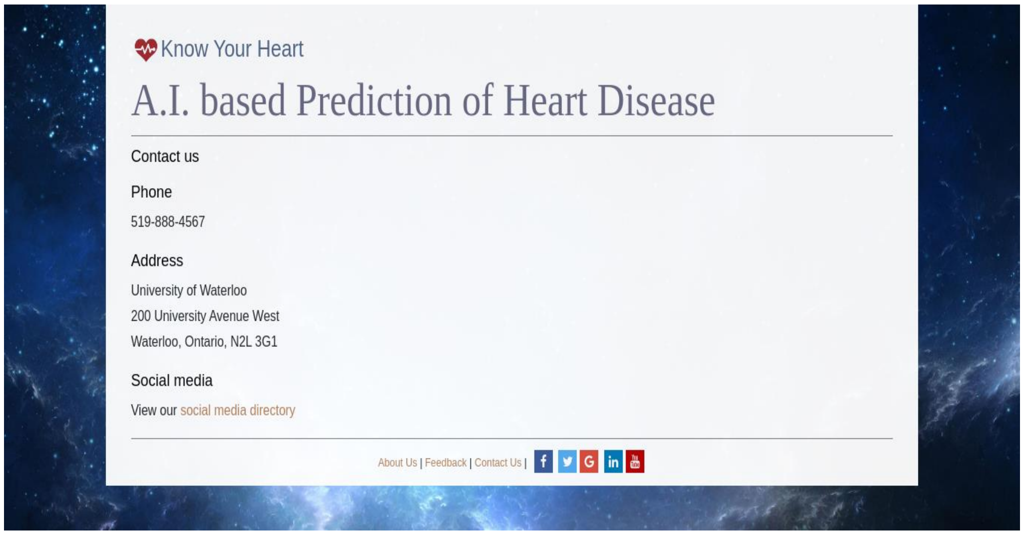
By clicking "Submit" I agree to be contacted at the email provided with more information about our work. I understand these texts may use computer-assisted technology.

Submit

Home

About Us | Feedback | Contact Us |

Contact Us page: This provides the details like contact number, address, etc. so that users can easily reach us.


Know Your Heart

A.I. based Prediction of Heart Disease

Contact us

Phone

519-888-4567

Address

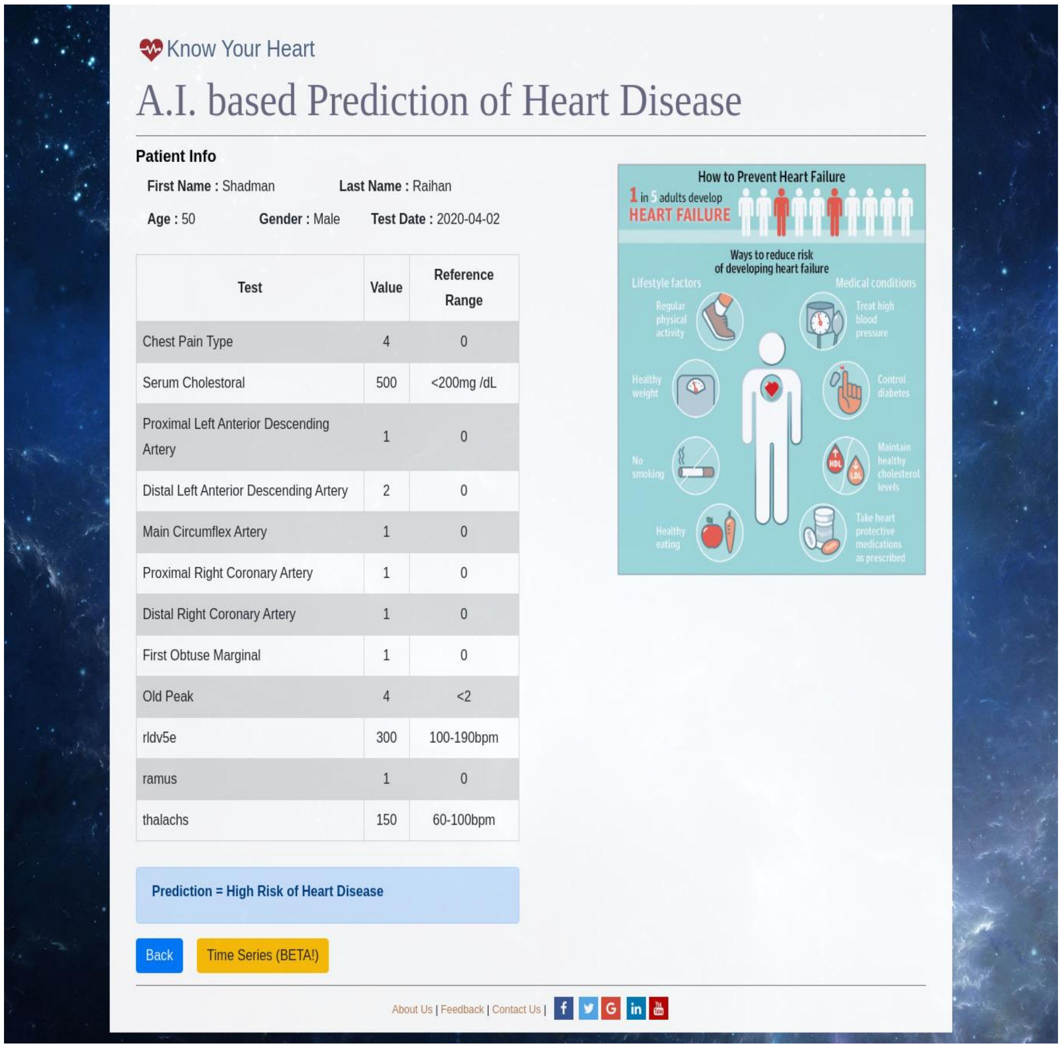
University of Waterloo
200 University Avenue West
Waterloo, Ontario, N2L 3G1

Social media

View our social media directory

About Us | Feedback | Contact Us |

2) **Result Page:** This page shows the results predicted by our website where there is High or Low risk of Heart Disease. The table shows the details of the patient and the normal range so that patient can compare his results. Also, there is a button "Time-Series BETA", by clicking it the user can open Time-Series Page.



Know Your Heart

A.I. based Prediction of Heart Disease

Patient Info

First Name : Shadman **Last Name :** Raihan

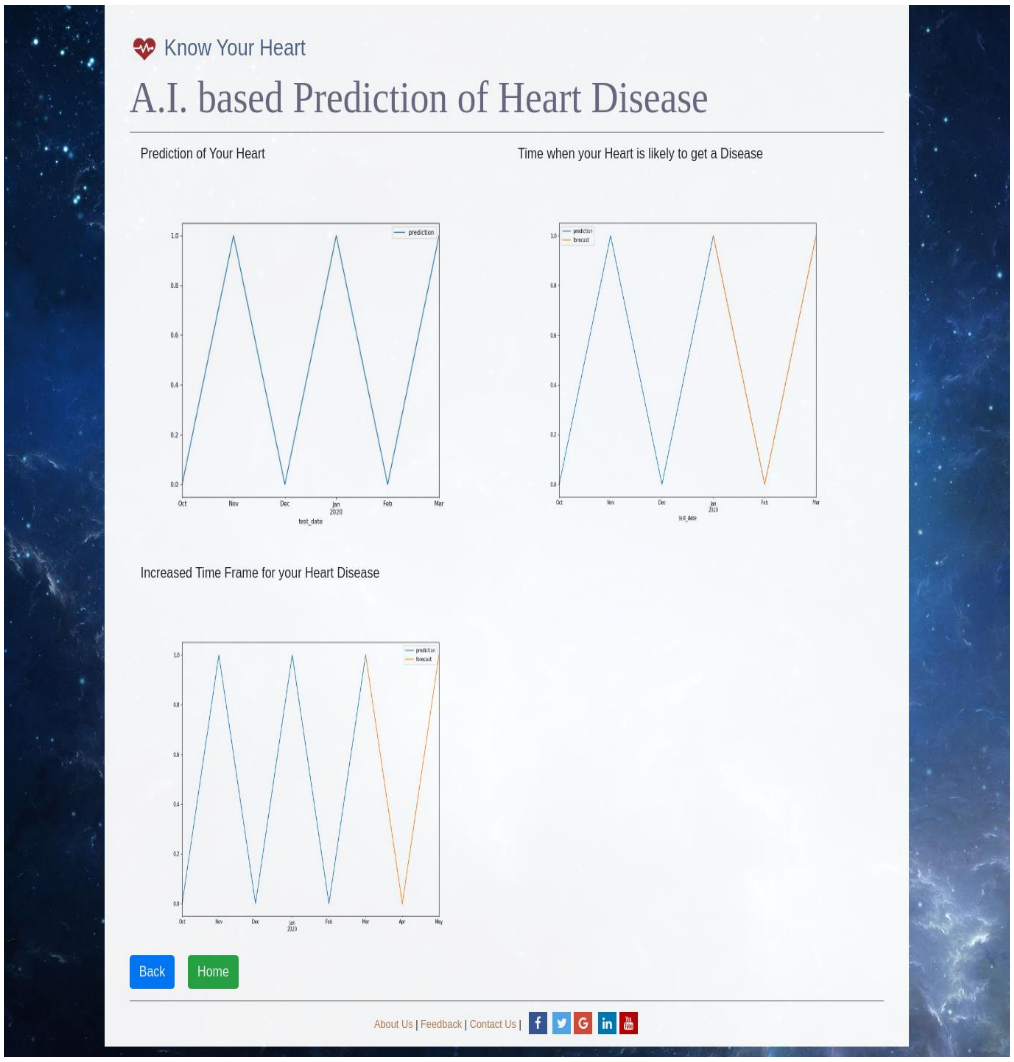
Age : 50 **Gender :** Male **Test Date :** 2020-04-02

Test	Value	Reference Range
Chest Pain Type	4	0
Serum Cholestoral	500	<200mg /dL
Proximal Left Anterior Descending Artery	1	0
Distal Left Anterior Descending Artery	2	0
Main Circumflex Artery	1	0
Proximal Right Coronary Artery	1	0
Distal Right Coronary Artery	1	0
First Obtuse Marginal	1	0
Old Peak	4	<2
rldv5e	300	100-190bpm
ramus	1	0
thalachs	150	60-100bpm

Prediction = High Risk of Heart Disease

Back
Time Series (BETA!)
How to Prevent Heart Failure
1 in 5 adults develop HEART FAILURE
Ways to reduce risk of developing heart failure
Lifestyle factors
Medical conditions
Regular physical activity
Treat high blood pressure
Healthy weight
Control diabetes
No smoking
Maintain healthy cholesterol levels
Healthy eating
Take heart protective medications as prescribed
About Us | Feedback | Contact Us |

3) **Time-Series Forecasting Page:** This page is under construction. For now, the graphs are generated by using dummy data of a patient. Basically, it provides the future predictions of a patient over a period of time like the time when your heart is likely to get a disease and an increased time frame of your heart disease.


Know Your Heart

A.I. based Prediction of Heart Disease

Prediction of Your Heart

Time when your Heart is likely to get a Disease

Increased Time Frame for your Heart Disease

TESTING

❖ Tools and technologies:

Used Pytest framework for unit testing→ a testing framework that allows us to write test codes using python. We can write code to test anything like database, API, even UI if we want. Basically, we used it for unit testing.
Used Selenium framework for integration testing→ a portable framework for testing web applications. Selenium provides a playback tool for authoring functional tests without the need to learn a test scripting language.
Used bandit for Security testing

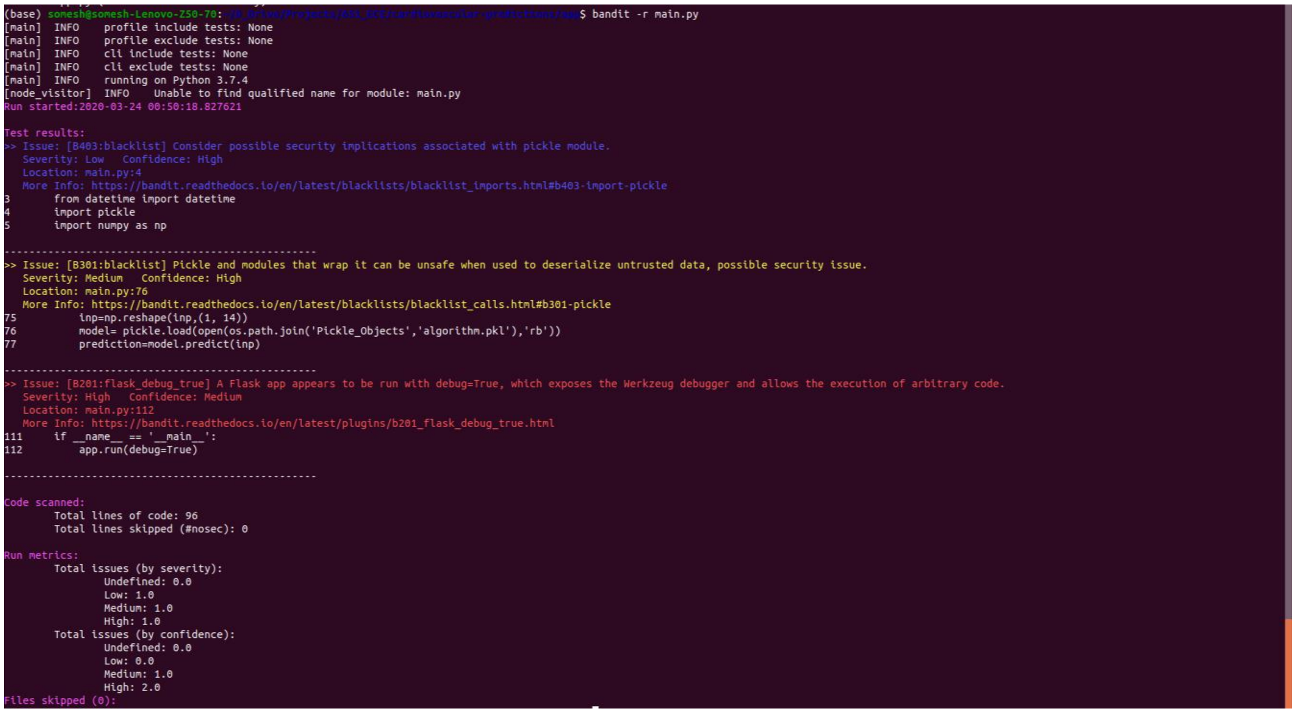
❖ Process Followed:

Unit Testing and Integration Testing

- Tests were written in the Pytest and mostly focused on the web application interface, using GET and POST requests to verify that the correct web page was served in response to the user request. These acted as unit tests for the various web application rendering functions and also as a system test ensuring that the application was up and able to take user input, as well as read and write to the database.
- For automated testing, we used the selenium framework.

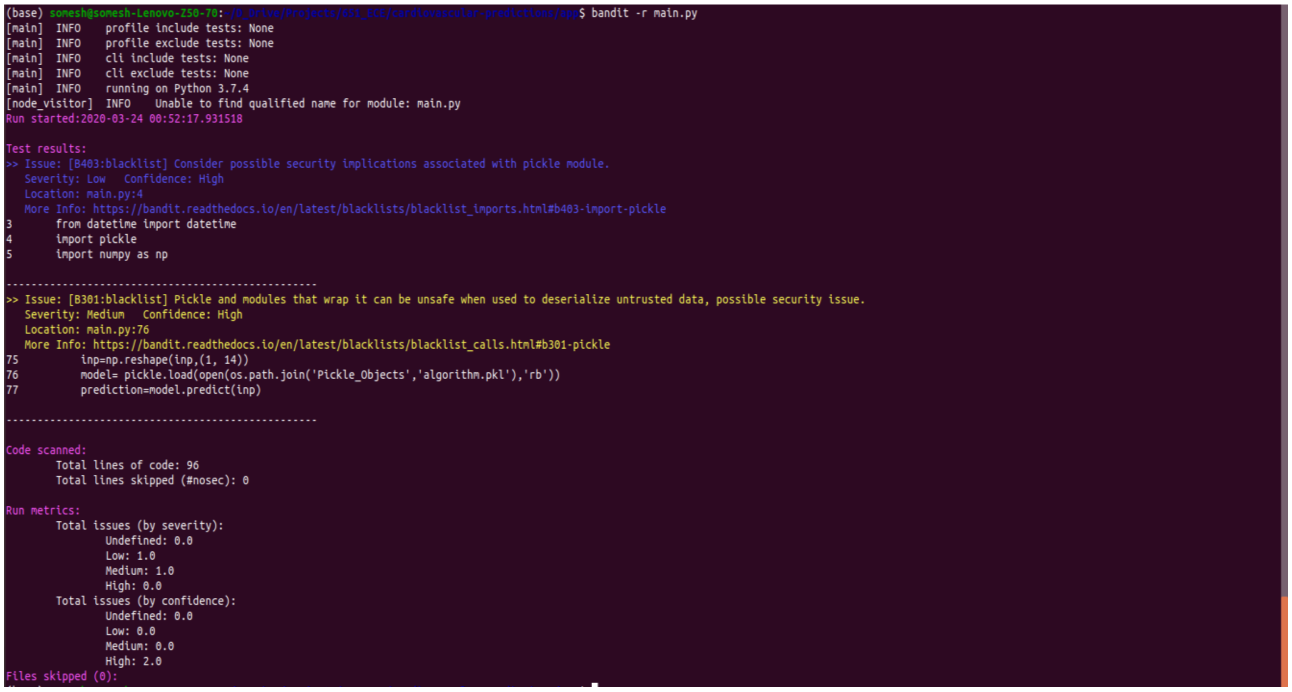
Security Testing

- For common security-related testing for our Website, we have used Bandit which is a python based tool that gives common security flaws in the software. Using this tool we were able to eliminate a few security-related issues that appeared while testing our code using Bandit and were able to successfully remove critical issues from our main code files.



```
(base) somesh@somesh-Lenovo-Z50-70:~/D_Drive/Projects/651_ECE/cardiovascular-predictions/app$ bandit -r main.py
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.7.4
[node_visitor]  INFO    Unable to find qualified name for module: main.py
Run started:2020-03-24 00:50:18.827621

Test results:
>> Issue: [B403:blacklist] Consider possible security implications associated with pickle module.
   Severity: Low   Confidence: High
   Location: main.py:4
   More Info: https://bandit.readthedocs.io/en/latest/blacklists/blacklist_imports.html#b403-import-pickle
3       from datetime import datetime
4       import pickle
5       import numpy as np

--------------------------------------------------
>> Issue: [B301:blacklist] Pickle and modules that wrap it can be unsafe when used to deserialize untrusted data, possible security issue.
   Severity: Medium   Confidence: High
   Location: main.py:76
   More Info: https://bandit.readthedocs.io/en/latest/blacklists/blacklist_calls.html#b301-pickle
75          inp=np.reshape(inp,(1, 14))
76          model= pickle.load(open(os.path.join('Pickle_Objects','algorithm.pkl'),'rb'))
77          prediction=model.predict(inp)

--------------------------------------------------
>> Issue: [B201:flask_debug_true] A Flask app appears to be run with debug=True, which exposes the Werkzeug debugger and allows the execution of arbitrary code.
   Severity: High   Confidence: Medium
   Location: main.py:112
   More Info: https://bandit.readthedocs.io/en/latest/plugins/b201_flask_debug_true.html
111     if __name__ == '__main__':
112         app.run(debug=True)

--------------------------------------------------

Code scanned:
        Total lines of code: 96
        Total lines skipped (#nosec): 0

Run metrics:
        Total issues (by severity):
                Undefined: 0.0
                Low: 1.0
                Medium: 1.0
                High: 1.0
        Total issues (by confidence):
                Undefined: 0.0
                Low: 0.0
                Medium: 1.0
                High: 2.0
Files skipped (0):
```

Severity of Risk: High



```
(base) somesh@somesh-Lenovo-Z50-70:~/D_Drive/Projects/651_ECE/cardiovascular-predictions/app$ bandit -r main.py
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.7.4
[node_visitor]  INFO    Unable to find qualified name for module: main.py
Run started:2020-03-24 00:52:17.931518

Test results:
>> Issue: [B403:blacklist] Consider possible security implications associated with pickle module.
   Severity: Low   Confidence: High
   Location: main.py:4
   More Info: https://bandit.readthedocs.io/en/latest/blacklists/blacklist_imports.html#b403-import-pickle
3       from datetime import datetime
4       import pickle
5       import numpy as np

--------------------------------------------------
>> Issue: [B301:blacklist] Pickle and modules that wrap it can be unsafe when used to deserialize untrusted data, possible security issue.
   Severity: Medium   Confidence: High
   Location: main.py:76
   More Info: https://bandit.readthedocs.io/en/latest/blacklists/blacklist_calls.html#b301-pickle
75          inp=np.reshape(inp,(1, 14))
76          model= pickle.load(open(os.path.join('Pickle_Objects','algorithm.pkl'),'rb'))
77          prediction=model.predict(inp)

--------------------------------------------------

Code scanned:
        Total lines of code: 96
        Total lines skipped (#nosec): 0

Run metrics:
        Total issues (by severity):
                Undefined: 0.0
                Low: 1.0
                Medium: 1.0
                High: 0.0
        Total issues (by confidence):
                Undefined: 0.0
                Low: 0.0
                Medium: 0.0
                High: 2.0
Files skipped (0):
```

Severity of Risk: Medium [Pickle and module that wrap it can be unsafe]
This issue would not affect our Website that much because for now, it is a small-scale project and we are using Pickle for just prediction. If in any worst-case any unknown activity happens, we will get notified.